NPS-MA-96-003

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Correlation in Multiversion Software

by

Toke Jayachandran

October 1996

Prepared for: Naval Postgraduate School
Monterey, CA 93943-5000

NAVAL POSTGRADUATE SCHOOL
MONTEREY, CA 93943

Rear Admiral M. J. Evans
Superintendent

Richard Elster
Provost

This report was prepared in conjunction with research conducted for the Naval Postgraduate School and funded by the Naval Postgraduate School.

This report was prepared by:

Toke Jayachandran
Professor, Department of Mathematics

Reviewed by:

WALTER M. WOODS
Chairman

Released by:

DAVID W. NETZER
Dean of Research

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 1996	Technical Report Jan - Sept. 96

4. TITLE AND SUBTITLE

CORRELATION IN MULTIVERSION SOFTWARE

5. FUNDING NUMBERS

N/A

6. AUTHOR(S)

TOKE JAYACHANDRAN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

NPS-MA-003

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this report qre those of the Author and do not reflect the official policy or position of the Department of Defense or the United States Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

It has been established both theoretically [1] and experimentally [2], that independently developed redundant software versions fail dependently. Several probability models that account for this phenomenon of concurrent failures have appeared in the literature. Tomek et al.,[3] proposed an intensity distribution that introduced a specific type of correlated failure pattern viz., pairwise correlation between software modules. They derived the intensity pmf for N = 2 and 3 modules and indicated the desirability of an efficient algorithm to compute the pmf for larger values of N. This paper contains an easily programmable algorithm to generate the pmf for any choice of N.

14. SUBJECT TERMS

redundant software, correlated failures, intensity distribution

15. NUMBER OF PAGES

11

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	

Correlation in Multiversion Software

Toke Jayachandran
Code MA/Jy
Naval Postgraduate School
Monterey, CA 93943

September, 1996

ABSTRACT

It has been established both theoretically [1] and experimentally [2], that independently developed redundant software versions fail *dependently.* Several probability models that account for this phenomenon of concurrent failures have appeared in the literature. Tomek *et al.*, [3] proposed an *intensity distribution* that introduced a specific type of correlated failure pattern viz., pairwise correlation between software modules. They derived the intensity pmf for N = 2 and 3 modules and indicated the desirability of an efficient algorithm to compute the pmf for larger values of N. This paper contains an easily programmable algorithm to generate the pmf for any choice of N.

1 INTRODUCTION

The two principal techniques for software redundancy are N-version programming [4] and the recovery blocks [5]. Both require multiple independently developed software versions to achieve high reliability in software systems. Initially, it was believed that failures in independently produced software occur independently; and system reliability computations were based on this premise. It was subsequently demonstrated, both theoretically [1] and experimentally [2] that multiple versions can fail simultaneously for some choices of inputs. As a result, reliability estimates assuming independent failures can be overly optimistic. Several papers introducing probability models that allow for concurrent failures have appeared in the literature recently. Nicola and Goyal [6] proposed a model for simultaneous failure of independent software modules and the model has been shown to provide a good fit to the experimental data in [2]. Tomek *et al.*, [3] introduced another model for generating the probability distribution (intensity distribution) of the number of modules (in an N-version system) that fail concurrently for a randomly selected input. The latter model incorporates the correlated failure syndrome into the intensity pmf through a parameter K that represents the probability that a pair of modules will produce identical outputs. They derived explicit

expressions for the pmf for N=2 and N=3 module software systems, and suggested that an efficient algorithm is needed to derive the pmf for larger values of N. This paper presents such an algorithm for generating the intensity pmf for different choices of the parameters N and K. The algorithm is easily programmable and is particularly suited for use with symbolic computation packages such as MAPLE©[1].

The Tomek *et al.*, [3] model for correlated failure is described in Section II and the algorithm for deriving the intensity pmf for chosen values of N and K is presented in Section III. A MAPLE program for generating the pmf and the output of the program for N=5 and K=.1 are included in the Appendix.

2 A PROBABILITY MODEL FOR CORRELATED FAILURES

Consider a redundant software system with N independently developed modules. Let $\Theta_N(X)$ be the proportion of modules (out of N) that fail (produce an incorrect output) for a randomly chosen input X. Then $\Theta_N(X)$ is a random variable assuming the values $\{0,\ 1/N, 2/N, \ldots, 1\}$. $\Theta_N(X)$ is called the intensity function and its probability distribution is referred to as the intensity distribution. For their probability based correlated failures model, Tomek *et al.* [3] assume that for each pair of modules, a proportion K of all possible inputs, will always generate identical outputs for the two modules. It is possible for two different pairs of modules to have identical inputs on two different sets of inputs, albeit the proportion of such inputs K is the same for all pairs. It is further assumed that a module will produce an incorrect output with probability p. For N=2 modules, the space of all possible inputs is comprised of two subsets R and its complement R'. R is the set of inputs for which the two modules will produce identical results, and for inputs from R' the module outputs are independent. The intensity function $\Theta_2(X)$ assumes the values 0, 1/2, 1 and

$$
\begin{cases}
Pr[\Theta_2(X)=0] = Pr[X\epsilon R].Pr[\text{both module outputs are correct}|X\epsilon R] \\
\qquad\qquad +Pr[X\epsilon R'].Pr[\text{both modules outputs are correct}|X\epsilon R'] \\
\qquad\qquad = K(1-p)+(1-K)(1-p)^2; \\
Pr[\Theta_2(X)=1/2] = Pr[X\epsilon R'].Pr[\text{exactly one output is correct}|X\epsilon R'] \\
\qquad\qquad = 2(1-K)p(1-p); \\
Pr[\Theta_2(X)=1] = Pr[X\epsilon R].P[\text{both module outputs are correct}|X\epsilon R] \\
+Pr[X\epsilon R'].P[\text{both module outputs are correct}|X\epsilon R'] \\
= Kp+(1-K)p^2;
\end{cases}
\tag{1}
$$

[1]MAPLE is a registered trademark of Waterloo Maple Software

In the case of $N = 3$ modules, the input space is partitioned into 3 types of subsets R_1, R_2, and R_3 where R_i (i = 2,3) is the set of inputs for which exactly i modules will produce identical results; R_1 is the set of inputs for which the module outputs are independent. There will be three subsets of the type R_2 and just one subset each of the types R_1 and R_3. The probabilities for the selection of an input from these subsets are K^2 for R_3, 3K(1 - K) for R_2 and $1 - K^2 - 3K(1-K) = (1-K)(1-2K)$ for R_1 and

$$Pr[\Theta_3(X) = j/3] = \Sigma_i Pr[X \epsilon R_i].Pr[\text{exactly j outputs are correct}|\, X \epsilon R_i] \quad j = 0 \ldots 3.$$

Therefore

$$\begin{cases} Pr[\Theta_3(X) = 0] = K^2(1-p) + 3K(1-K)(1-p)^2 + (1-K)(1-2K)(1-p)^3; \\ Pr[\Theta_3(X) = 1/3] = = K^2.0 + 3K(1-K)p(1-p)^2 + (1-K)(1-2K)p(1-p)^2; \\ Pr[\Theta_3(X) = 2/3] = = K^2.0 + 3K(1-K)p(1-p) + (1-K)(1-2K)p^2(1-p); \\ Pr[\Theta_3(X) = 1] = = K^2 p + 3K(1-K)p^2 + (1-K)(1-2K)p^3. \end{cases} \tag{2}$$

The calculation of the probabilities of selecting an input from the subsets partitioning the input space, and the conditional pmf of $\Theta_N(X)$ becomes increasingly more difficult as the number of modules N increases. An efficient algorithm that will perform the needed book keeping in a systematic fashion is presented in the next section.

3 AN ALGORITHM FOR GENERATING THE INTENSITY DISTRIBUTION

For an N module software system, the input space is partitioned in N types of subsets R_i, $i = 1, 2, \ldots, N$. Inputs from subset type R_i will result in identical outputs from i of the N modules. The number of subsets of type R_i, except for type R_1, is equal to $\binom{N}{i}$ the number of different ways of selecting i modules from the available N modules. There is just one subset of type R_1 and the module outputs are independent for inputs from this subset. The table below illustrates the pattern for the conditional probabilities $Pr[\Theta_N(X) = j/N|\, X \epsilon R_i]$ when N = 5.

TABLE 1

j/5 =	0	1/5	2/5	3/5	4/5	1
R_5	$(1-p)$	0	0	0	0	p
R_4	$(1-p)^2$	$p(1-p)$	0	0	$p(1-p)$	p^2
R_3	$(1-p)^3$	$2p(1-p)^2$	$p^2(1-p)$	$p(1-p)^2$	$2p^2(1-p)$	p^3
R_2	$(1-p)^4$	$3p(1-p)^3$	$3p^2(1-p)^2+p(1-p)^3$	$p^3(1-p)+3p^2(1-p)^2$	$3p^3(1-p)$	p^4
R_1	$(1-p)^5$	$5p(1-p)^4$	$10p(1-p)$	$10p^3(1-p)^2$	$5p^4(1-p)$	p^5

The probability entries in the table constitute a 5 x 6 matrix P which can be expressed as the sum A + B of the two triangular matrices

$$A = \begin{bmatrix} (1-p) & 0 & 0 & 0 & 0 & 0 \\ (1-p)^2 & p(1-p) & 0 & 0 & 0 & 0 \\ (1-p)^3 & 2p(1-p)^2 & p^2(1-p) & 0 & 0 & 0 \\ (1-p)^4 & 3p(1-p)^3 & 3p^2(1-p)^2 & p^3(1-p) & 0 & 0 \\ (1-p)^5 & 5p(1-p)^4 & 10p^2(1-p)^3 & 10p^3(1-p)^2 & 5p^4(1-p) & p^5 \end{bmatrix}$$

and

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & p \\ 0 & 0 & 0 & 0 & p(1-p) & p^2 \\ 0 & 0 & 0 & p(1-p)^2 & 2p^2(1-p) & p^3 \\ 0 & 0 & p(1-p)^3 & 3p^2(1-p)^2 & 3p^3(1-p) & p^4 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (3)$$

The above pattern persists for all N and the two $N \times N$ +1 matrices, in the general case, have the form $A = (a_{ij})$ and $B = (b_{ij})$ where

$$a_{ij} = \begin{cases} \binom{i-1}{j-1} \; p^{(j-1)} \; (1-p)^{(i-j+1)} & \text{for} \quad j > i = 1, 2, \ldots, N-1 \\ \binom{N}{j-1} \; p^{(j-1)} \; (1-p)^{(n-j+1)} & \text{for} \quad i = N \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$$b_{ij} = \begin{cases} \binom{i-1}{N+1-j} \; p^{i+j-N-1} \; (1-p)^{N+1-j} & \text{for} \quad j > N-i = 1, 2, \ldots, N-1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The entries in the matrix $P = (a_{ij} + b_{ij})$ are the conditional probabilities $Pr[\Theta_N = (j-1)/N | X \epsilon R_i], i = 1, \ldots, N$ and $j = 1, \ldots, N+1$. The unconditional probabilities or the intensity distribution is obtained by multiplying the matrix P on the left by the 1-row matrix $Q = [q_1, q_2, \ldots, q_n]$ where

$$q_i = \binom{N}{i} K^{N-i-1}(1-K)^i \quad \text{for} \quad i = 1, 2, \ldots, N-1 \tag{6}$$

$$q_N = 1 - \sum_i q_i. \tag{7}$$

Note that Q is just the vector of probabilities for an input to be in each of the subset types $R_N, R_{N-1}, \ldots, R_1$.

The algorithm for computing the intensity pmf of $\Theta_N(X)$ can be described by the following 3-step process. For specified values of the parameters N and K

1. Determine the 1-row matrix Q of input probabilities.

2. Evaluate the matrix $P = A + B$, the matrix of conditional probabilities, $Pr[\Theta_N = (j-1)/N | X \epsilon R_i]$.

3. Compute the matrix product Q $\times$ P which is a 1 $\times$ N + 1 matrix to obtain the intensity pmf of $\Theta_N(X)$.

APPENDIX

Three MAPLE procedures r(N,K), Q(N,K) and P(N,K) to generate q_N the matrix Q in (6) and (7) and the matrix P = A + B in (3) are shown below. A printout of the MAPLE output creating these procedures and the computational results for N =5 and K = .1 is also included.

```
r: = (N,K) - > 1 - sum ((binomial(n,i))* (K^(N-1-i))*(1 - K)^ i), i = 0..N-2);

Q: = (N,K) - > array ([(seq(binomial(N,i)*K^(N -1-i)*(1 - K)^ i, i = 0..n-2), r(n,k))]);

P: = proc(N,K) local A,B,C,s,t;
A: = array (1..N, 1..N+1):
for s to N do
for t to N + 1 do
if s < N then
if s > t - 1 then
A[s,t]: = (binomial(s - 1, t - 1)* (K^(t-1))* (1 -K)^(s - t -1))
else A[s,t]: = 0 fi;
else A[s,t]: = (binomial(n,t-1)*(K^(t-1))* (1 - K)^(N-t-1)); fi; od;
od;
B: = array(1..N,1..N + 1):
for s to N do
for t to N + 1 do
if s < N then
if t > N - s + 1 then
B[s,t]: = (binomial(s - 1,N+1 - t)* K^(s + t-N-1)* ((1 - K)^(N +1 -t))
else B[s,t]: = 0 fi;
else B[s,t]: = 0 fi; od;
od;
C: = evalm (A + B);
end;
```

Finally, the MAPLE expression

```
evalm (Q(N,K)&* P(N,K));
```

will display the desired intensity pmf.

```
> r:=(N,K)->1-sum((binomial(N,i))*(K^(N-1-i))*((1-K)^i),i=0..N-2);
```

$$r := (N, K) \to 1 - \left(\sum_{i=0}^{N-2} \text{binomial}(N, i)\, K^{(N-1-i)}\, (1-K)^{i} \right)$$

```
> Q:=(N,K)->array([(seq(binomial(N,i)*K^(N-1-i)*(1-K)^i,i=0..N-2),r(
  N,K))]);
```

$$Q := (N, K) \to \text{array}([\,\text{seq}(\text{binomial}(N, i)\, K^{(N-1-i)}\, (1-K)^{i}, i = 0 \,..\, N-2), \text{r}(N, K)])$$

```
> P:=proc(N,K) local A,B,C,s,t;
> A:=array(1..N,1..N+1):
  for s to N do
> for t to N+1 do
  if s<N then
> if s>t-1 then
  A[s,t]:=(binomial(s-1,t-1))*(K^(t-1))*((1-K)^(s-t+1))
> else A[s,t]:=0 fi;
> else A[s,t]:=(binomial(N,t-1))*(K^(t-1))*((1-K)^(N-t+1)); fi; od;
> od;
> B:=array(1..N,1..N+1):
> for s to N do
> for t to N+1 do
  if s<N then
> if t>N-s+1 then
  B[s,t]:=(binomial(s-1,N+1-t))*(K^(s+t-N-1))*((1-K)^(N+1-t))
> else B[s,t]:=0 fi;
> else B[s,t]:=0; fi; od;
> od;
  C:=evalm(A+B);
  end;
```

```
P := proc(N, K)
local A, B, C, s, t;
    A := array(1 .. N, 1 .. N + 1);
    for s to N do for t to N + 1 do
            if s < N then
                if t - 1 < s then
                    A[s, t] := binomial(s - 1, t - 1)*K^(t - 1)*(1 - K)^(s - t + 1)
                else A[s, t] := 0
                fi
            else A[s, t] := binomial(N, t - 1)*K^(t - 1)*(1 - K)^(N - t + 1)
            fi
        od
    od;
    B := array(1 .. N, 1 .. N + 1);
    for s to N do for t to N + 1 do
            if s < N then
                if N - s + 1 < t then B[s, t] :=
                    binomial(s - 1, N - t + 1)*K^(s + t - N - 1)*(1 - K)^(N - t + 1)
                else B[s, t] := 0
                fi
```

```
            else B[s, t] := 0
            fi
        od
    od;
    C := evalm(A + B)
end
```

```
> r(5,.1);
```

$$.1854000000$$

```
> Q(5,.1);
```

$$[.0001, .0045, .0810, .7290, .1854000000]$$

```
> P(5,.1);
```

$$\begin{bmatrix} .9 & 0 & 0 & 0 & 0 & .1 \\ .81 & .09 & 0 & 0 & .09 & .01 \\ .729 & .162 & .009 & .081 & .018 & .001 \\ .6561 & .2187 & .0972 & .0252 & .0027 & .0001 \\ .59049 & .32805 & .07290 & .00810 & .00045 & .00001 \end{bmatrix}$$

```
> evalm(Q(5,.1) &* P(5,.1));
```

$$[.6505577460, .2337797700, .08510346000, .02643354000, .003914730000, .0002107540000]$$

```
>
```

REFERENCES

[1]. D. E. Eckhardt and L. D. Lee, "A Theoretical Basis for the Analysis of Multiversion Software Subject to Coincident Errors," IEEE Trans. Software Eng., Vol. SE-11, pp. 1511-1517, Dec. 1985.

[2]. J. Knight and N. Leveson, An Experimental Evaluation of the Assumption of Independence in Multiversion programming," IEEE Trans. Software Eng., Vol. SE-12, pp. 96-109, Jan. 1986.

[3]. L. A. Tomek, J. K. Muppala and K. S. Trivedi, " Modeling Correlation in Software Recovery Blocks, IEEE Trans. Software Eng., Vol. SE-19, pp. 1071-1086, Nov. 1993.

[4]. A. Avizienis, "The N-Version Approach to Fault-Tolerant Software," IEEE Trans. Software Eng., Vol. SE-11, pp. 1491-1501, Dec. 1985.

[5]. B. Randell , "System Structure for Software Fault Tolerance," IEEE Trans. Software Eng., Vol. SE-1, pp. 220-232, Jun. 1975.

[6]. V. F. Nicola and A. Goyal, "Modeling of Correlated Failures and Community Error Recovery," IEEE Trans. Software Eng., Vol. SE-16, pp. 350-359, Mar. 1990.

Distribution List

	No. of copies
Director Defense Technology Information Center Cameron Station Alexandria, VA 22314	2
Research Office Code 81 Naval Postgraduate School Monterey, CA 93943	1
Library Code 52 Naval Postgraduate School Monterey, CA 93943	2
Professor Toke Jayachandran Department of Mathematics Naval Postgraduate School Monterey, CA 93943	5
Chairman, W. Max Woods Department of Mathematics Naval Postgraduate School Monterey, CA 93943	2